Attorney Docket No. P5795
Client Matter No. 80168-0240

UNITED STATES PATENT APPLICATION

OF

LEONID KHODULEV,

ALEXANDRE KRAVTCHENKO

AND

ANDREI SKALDIN

FOR

AN IMPORT/EXPORT UTILITY AND A METHOD OF PROCESSING DATA USING THE SAME

BACKGROUND OF THE INVENTION

Field of the Invention

[0001] The present invention relates to an import/export utility, and more particularly, to an import/export utility that imports and exports business objects.

Discussion of the Related Art

[0002] Conventionally, a database loader is used to load information to E-commerce system. However, the database loader loads only tables.

[0003] In E-commerce system, there are business objects (BO) which are resources. The business objects include information regarding payments, units, and credit cards, for example. Therefore, there is a need for a utility that is capable of receiving/sending information from/to the business object or restoring the business object on the other system.

SUMMARY OF THE INVENTION

[0004] Accordingly, the present invention is directed to an Import/Export Utility that substantially obviates one or more of the problems due to limitations and disadvantages of the related art.

[0005] Additional features and advantages of the invention will be set forth in the description which follows, and in part will be apparent from the description, or may be learned by practice of the invention. The objectives and other advantages of the invention will be realized and attained by the structure particularly pointed out in the written description and claims hereof as well as the appended drawings.

0006] To achieve these and other advantages and in accordance with the purpose of the present invention, as embodied and broadly described, a method of processing data in a system including an utility, includes the steps of starting a session, selecting a file on a local drive or by URL, wherein the file includes a name of a business object, uploading the file including the name of a business object to a server, storing data of the file in a database of the utility, performing asynchronous data processing, and downloading and saving a report after the data processing is completed.

[0007] It is to be understood that both the foregoing general description and the following detailed description are exemplary and explanatory and are intended to provide further explanation of the invention as claimed.

BRIEF DESCRIPTION OF THE DRAWINGS

[0008] The accompanying drawings, which are included to provide further understanding of the invention and are incorporated in and constitute a part of this specification, illustrate embodiments of the invention and together with the description serve to explain the principles of the invention. In the drawings:

[0009] FIG. 1 is a block diagram of data processing according to the present invention;

FIG. 2 is a block diagram illustrating ImpExp database;

FIG. 3 is a flow chart describing the data processing according to the present invention; and

FIG. 4 is a block diagram of data processing for an export operation according to the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0010] Reference will now be made in detail to the preferred embodiment of the present invention, examples of which are illustrated in the accompanying drawings.

[0011] The present invention will now be described with reference to FIG. 1.

[0012] The present invention includes a system having a client, a server, and an import/export utility. The user sends requests to the server and receives responses from the server. The client uses an internet browser or a system console to work with the import/export utility through a graphical user interface (GUI) or a command line interface, respectively. The import/export utility is designed to process the import file on the remote server or the client. The import/export utility imports/exports data from/to external files, by invoking business objects. The import/export utility receives information from the import file or the business objects, and loads the information into business objects or restores the information on another system. The business objects may be written in Java, for example, and may contain resources in E-commerce systems, such as payments, units, credit card, and user information, for example.

[0013] For a successful interaction with the import/export utility, the business object should be described in a data objects repository (DOR). The business object is implemented with three static methods - doImportExport, findByAttributes, and

close. The doImportExport is a method to perform a particular import/export command. A name of operation and two lists of parameters are passed to the doImportExport. The business object is responsible for validation of the operation. The second parameter list in the doImportExport may be null since it is important only for updating the operations. The last parameter in each of the lists is an interface to support export operation. The doImportExport returns null if the operation was successful and an error string if the operation was not successful. If the error string is returned, then the returned error string is put directly into a discard file that will be returned to the user. The findByAttributes supports object references in import files. The findByAttributes receives a list of attributes of the validated object and returns unique object identifier, such as GUID (Globally Unique Identifier), for example, or null if error occurs. The close method notifies the end of the process.

[0014] FIG. 3 shows the steps of import/export session of the system according to the present invention.

[0015] In step 21, the session is started. In this step, a unique session ID is generated and configuration settings are checked. The user can pass several parameters in this step – a first parameter being 'user Name' which is used in session ID generation, a second parameter being 'userPass' which is not used in this step, and a third parameter being 'fileNum' which determines how many 'FILE' controls will be placed in a 'SelectFile' screen.

0016] In step 22, an import file is selected by browsing local drives or typing URL. The 'FILE' control is used in browser interface, and a command line parameter is used for a list of files to be uploaded in command line interface.

[0017] In step 23, the import file is uploaded to a server via the browser or the command line interface.

[0018] In step 24, the data of the import file is stored in a import/export utility database, ImpExp database. The import/export database includes six tables, as shown in FIG. 2, – "IE_UID" for storing user names and corresponding session IDs; "IE_STATUS" for storing current session status which is periodically updated; "IE_IODATA" for storing initial session data obtained from the import file; "IE_EXPDATA" for storing results of the export operations; "IE_IOERRORS" for storing error messages; and "IE_LOG" for storing all sessions log. Then, asynchronous data processing begins in step 25.

[0019] In step 26, reading and validating of the data is performed and errors are logged. Further, the doImportExport method provided by the business object is invoked.

[0020] In step 27, status of the data processing is periodically displayed.

[0021] In step 28, downloading and saving report is performed after the data processing is finished.

[0022] The user on the client side can control the process using a SelectFile screen, Start screen, Status screen, GetReport screen, and Abort screen. In the SelectFile

screen, the user receives unique session ID and the user can stop the session and resume the session later. In the SelectFile screen, the user can browse local disks or input URL or the network path to select the file to be uploaded. When the import file was successfully uploaded and loaded into the ImpExp database, the Start screen appears. In the Start screen, the user can start the process or postpone the process. The Status screen appears after the process is initiated. The Status screen displays a current business object that is being processed, operation that is being performed, a sequence number of the current attribute section in current control section, and a number of the current data line in the current attribute section. The user can terminate the session and resume the session later. However, the termination will not be performed until processing of the current data line is completed.

[0023] After successful completion of the process, the user receives the GetReport screen. In the GetReport screen, the user can initiate downloading of the discard file and log files. Further, the user can select session data to be removed from the ImpExp database and is able to choose whether to download the whole discard file or only lines where errors exist. Also, the user receives an Abort screen when the user chooses to stop the process on the Status screen, and an Error screen when fatal error occurs that makes it impossible to continue the whole session.

[0024] The import file includes 3 types of data – a name of business object to be invoked, operation to be performed, such as insert new user and delete upgrade, for example, and properties of the business object. A simplified CSV (comma-separated

values) format may be used for the import file. The import file is a text file of several lines. In the import file, empty lines and comment lines can be inserted anywhere, and all other lines are separated into three groups – control section or command lines, attribute names or header lines, and data or body lines. The command line, containing operation and object names, starts with "<" and defines a command for all subsequent body lines. The header line, containing a list of the columns in the body lines, starts with "^" and defines the format for the body lines. The body line, which is a list of values (usually business object attributes: text, numeric, data, for example), contains one value for each of the columns from the last header line. Operand in lines are comma-separated and space symbols between operands are non-significant. For some operations, such as update, for example, it might be necessary to pass two values for the each of the columns, for example, old and new values of the column, to the business object. In this case, those values are separated with a special character "|" and space symbols are non-significant. The following is an example of the import file structure:


```
### comment; command line follows:
< command1 , className1
# new group of body lines starts after a header line:
^ columnName1, columnName2, columnName3, . . .
value11, value12, value13, . . .
value21, value22New | value22Old, value23, . . .
```

next group starts here:

^ columnName4, columnName5, . . .

value31, value32, . . .

value 41, value42, . . .

. . .

next command starts

< command2, className2

[0025] The body lines of the import file can contain references to the other business objects. The references are used to obtain the properties, GUID for example, of the referenced object while the main object importing is being processed. The following is an example of a reference:

[0026] [refObjName objAttrName 1 = "objAttrVal1" objAttrName2 = "objattrVal2"...]

The references can be nested. The following is an example of a nested reference:

[User Name = "Alex" Group = [UserGroup Name = "software"]]

[0027] However, a reference to a business object being referred to has to be created before a business object with a body line containing the reference is created.

[0028] The default values for some of the object attributes may be defined in the control section. The definition strings are separated from object and operation fields and from each other by commas – for example,

< operation, objName, orgName=AOL, orgType=SELLER

[0029] FIG. 4 shows data processing in the system performing an export operation according to the present invention.

[0030] For an export operation, IExport interface (5) is passed to the business object (3) to support the export operation, in addition to the steps of the import/export session explained above. The IExport interface includes startExport, onReceiveExport, and stopExport methods and those methods are called from the business object.

[0031] The startExport signals starting of the export data process and forms the proper control section in an output file (6) of the operation. The startExport contains two parameters - operation which is a name of the operation to be inserted into a control section of output file, and className which is a business object class name to be inserted into the control section of the output file. The onReceiveExport calls back for the export data process and throws one attribute set at a time. The onReceiveExport contains two parameters - errMsg which is an error message for the operation, and attributes which is a single attribute set returned by the business object for the operation. If the errMsg is not null, than the operation is cancelled. The stopExport method signals stopping of the export data process. Then, results of the successfully ended operation is put in the ImpExp database.

[0032] The business object can export all of its data or part of its data. In the case where the business object exports all of its data, the following syntax of the import file can be used, for example:

< read, BusObj

^ *

In the case where the business object exports part of its data, the following syntax of the import file can be used, for example:

< read, BusObj

^attrName1, attrName2

val1, val2

In this example, only the objects for which attrName1 = val1 and attrName2 = val2 will be exported.

[0033] The following is another example for the export operation with following syntax in the import file:

command line

< Insert, User

header line - enumerated attributes

^ name, age, group

instances

Ivan, 23, [Group name = "Software"]

Alex, 30, [Group name = "Software"]

[0034] In this example, two instances of business object "User" with listed attributes are exported. The import/export utility passes the import file and receives a business object name (resource name).

[0035] Further, Authentication control and Session Management Utility (SMU) can be implemented in the import/export utility. With the Session Management Utility, the user can select one or several sessions to manage and resume the processing of the selected sessions, to stop processing active sessions, to remove all selected sessions, to except active sessions from the ImpExp internal database, and to check properties of the selected sessions. The Session Management Utility can be started at the Select File screen.

[0036] In the present invention, the import/export utility is like copying data from the business object. The import/export utility knows nothing about which E-commerce system it is working with. The real work is done inside of the business object itself. The import/export utility is a deliverer of information. The import/export utility delivers the data from the external files to the business object and vice versa, and the business object performs the validation and performs the tasks, such as adding, deleting, or updating of the data. The data can be changed without changing the business object. Further, the import/export utility of the present invention provides a process monitoring function, an error handling function, and a reporting function.

[0037] It will be apparent to those skilled in the art that various modifications and variations can be made in the import/export utility of the present invention without departing from the spirit or scope of the invention. Thus, it is intended that the

present invention covers the modifications and variations of this invention provided that they come within the scope of any claims and their equivalents.